Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Environmental Sciences 12 (2012) 568 – 575

Procedia Environmental Sciences

2011 International Conference on Environmental Science and Engineering

(ICESE 2011)

Contract Performing Scenario Based Services Collaborative Self-evolution Model*

Denghui Zhang[a],Ji Gao[b],Bin Xie[c]

[a]*College of Information and Technology,Zhejiang Shuren University*
Hangzhou, Zhejiang Province, China,zzddhh@263.net
[b]*College of Computer Science,Zhejiang University*
Hangzhou, Zhejiang Province, China,gaoji1@zju.edu.cn
[c]*Institute of Remote Sensing and Earth Sciences,Hangzhou Normal University*
Hangzhou, Zhejiang Province, China,xiebinbingdiao@gmail.com

Abstract

Self-organized service chain formed by service agent can't complete the scheduled collaborative target because of exceptions of service provider. This paper presents a service chain evolution model based on collaborative state. In the model, service collaborative relationship is modeled as a contract set, whose performance results are regarded as the result of service coordination state. And then the monitoring and evolution policies of collaborative process are established based on collaborative state. On this basis, a collaborative scenario-driven adaptive and self-evolutionary mechanism for service chain is established. Based on planning policies, different evolution plans are formulated to obtain a better flexibility of service chain.

*Keywords:*virtual organization; service exception; collaborative scenario; self-evolution

1.Introduction

Currently, using service-oriented architecture (SOA) [1] to build collaborative service based virtual organization (VO) is becoming an effective method to solve resource sharing and collaborative problems

* This work is partially supported by National High-Technology Research and Development Program (863) of China (Grant #2009AA12Z212) to B. Xie and Priority Theme Emphases Project of Zhejiang Province, China (Grant #2010C11045) to D. Zhang.

 doi:10.1016/j.proenv.2012.01.319

[2-3]. But, service collaboration can't effetely, flexibly cope with dynamic changes of the network environment and application requirements because of the non-autonomous of software services. For this purpose, researchers have studied methods of service workflow adaptive evolution, dynamic formation of virtual organizations, dynamic collaboration of multi-Agent and etc. for the service collaborative change [4-12].

However, these collaborative evolution models are difficult to apply to the open, dynamic, heterogeneous network environment. On the one hand, the operational environment of VO is lack of clear semantic description. On the other hand, the evolution of VO members is lack of guide policy. Fortunately, policy-based self-management [13] and norm-based monitoring [14] have separately been well studied in different areas. Combining with contract and policy, this paper proposes a contract performing scenario-driven VO self-evolution model, which considers the VO operational process as VO members performing collaborative activities in accordance with contractual agreements.

2.Overview of the model

Contract performing circumstance driven adaptive and flexible evolution model defined as the 5-tuple is shown in Figure 1.

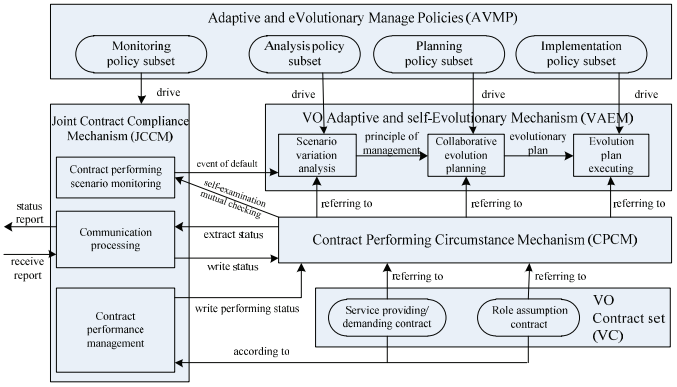

Fig. 1 VO adaptive and flexible self-evolution model

CCAE=(AVMP, VC, CPCM, JCCM, VAEM)

AVMP: Adaptive and eVolutionary Manage Policies;

VC: VO Contract set, which includes service providing/demanding contract and role assumption contract;

CPCM: Contract Performing Circumstance Mechanism, which reflects the implementation status of service performing contract, is used to compose service collaborative scene of VO runtime;

JCCM: Joint Contract Compliance Mechanism. It respects contract performing agreement by signing service contracts, and monitors the implementation process of collaborative services in VO through the self-examination and mutual checking of contract performing status;

VAEM: VO Adaptive and self-Evolutionary Mechanism, which includes four phases: scenario monitoring, scenario variation analysis, collaborative evolution planning, evolution plan executing, is used to support adaptive and flexible evolution for services collaboration.

3.Formal Specifications

3.1.Adaptive and evolutionary manage policies (AVMP)

Policy is a set of rules, in which every rule is composed of an evaluation condition ρ and an action (or an action sequence) α. The action α is executed by agent while agent believes that the evaluation condition is true. Formally, a policy can be defined in the form as follows.

Definition 1 Policy

policy={(ρi,αi)|i∈{1,…,m}} ,In this definition,ρi represents an evaluation condition of this policy, and αi represents the action sequence executed after that ρi is satisfied, αi can defined as {ai1,…,ain}.

The execution process of a policy is an action sequence, formally defined as follows.

Definition 2 Action Sequence of Policy Execution

PolicyExecute(policy)=$\bigcup_{i=1}^{m}$ (ρi $\Rightarrow$ Does(ain))?,ain∈αi

Wherein, ';' and '?' is the primitive symbols of dynamic description logic, ';' denotes the action sequence, '?' denotes action test.

Therefore, PolicyExecute (policy) denotes the action sequence of policy execution, that is, the execution process of policy is represented by the execution of an action sequence. For every rule in the policy, agent judges whether the evaluation condition of the rule is satisfied, and execute the action of the rule if satisfied.

3.2.VO contract set (VC)

Contract is the conditions and basis of collaboration between VO members. It is divided into two categories: role assumption contract and service providing/demanding contract. Role assumption contract is the credentials of the service to join VO, and service providing/demanding contract represents rights and obligations between cooperation partners.

- Role assumption contract: This contract is defined as the 4-tuple: BPR = (PRR, PIR, Right, Obligation). PRR denotes the response role set of role assumption. PIR is the starting role set of role assumption. Right expressed by a group of norm sets denotes the right of the role. Here, each norm provides the trigger concept, the effective time period and the specific disposal operations. Obligation denotes the obligations of the role, which is also expressed by a group of norm sets.
- Service providing/demanding contract: Contract of service(SC) is defined as a 3-tuple: SC=(CBM,CSI,PN), where CBM denotes contract's basic information such as contract number, credit card number, service provider code and so on. CSI denotes the detailed service items provided by contract. PN is the half-ordered set of performing norms of SC.

3.3.Contract performing circumstance mechanism (CPCM)

Definition 3 Performing Norm of Service Contract

PN=OB_a^{SC} (ρ≤δ| σ) |FB_a^{SC} (ρ≤δ| σ)| PB_a^{SC} (ρ≤δ| σ), indicating respectively that, when σ holds true, the role a (contractors of SC)is obligated to, forbidden to, or authorized to make ρ true before deadline δ (here ρ, δ, and σ are all the propositions describing service cooperation status).

Definition 4 Executing Status of Contract Performing Norm

ES = (PN-number, Status-type, Status-description), where PN-number denotes the number of current performing PN that belongs to SC. Status-type denotes the type of performing states, such as success, failure and exception etc. Status-description gives the description of performing scenario.

When performing norm (PN) of obligation type (OB_a^{SC}) or authorization type (PB_a^{SC}) executes successfully, the Status-description is the example of proposition ρ or ρ, which is specified by PN that should be transformed into true, otherwise, failure or exception description including description type and content will be given. Performing norm (PN) of forbidden type (FB_a^{SC}) would not perform or record implementation status under normal circumstances, until encountering the breach of contract. Here, the type of implementation status is indicated by "abnormal" and exception description is given.

Based on the above definitions of SC and PN, Contract performing circumstance (CPCsc) of each SC is modeled as the sequence (execution orders are specified by contact performing agreement) of executing status (ESi) of contract performing norm.

CPCsc = {ES1, ES2, …, ESm}, ESi = Executing-state (PNj), PNj (∈PN-setsc), where, PN-setsc denotes performing norm set constituted for service contract. PNj represents one of the performing norms. Executing-state (PNj) denotes implementation status of performing norm.

3.4.Joint contract compliance mechanism (JCCM)

Joint Contract Compliance Mechanism (JCCM), which is used to implement joint compliance for single contract, is expressed as the following 7-tuple:

JCCM={VM-set,Contract-set,PN-set,Self-executing,Self-examining,Inter-reporting,Inter-examining}

VM-set: the set of members in the VO;

Contract-set: the set of service contracts in VO;

PN-set: Joint performing norm set of all service contracts, pns(sc1)∪pns(sc2), …,∪pns(scn), where, pns(sci) denotes the performing norm set of sci (∈Contract-set);

Self-executing: Vm-set × Contract-set □Ppn-set. Based on the signed sc (∈ Contract-set), each vm (∈ VM -set) executes service contract performing norm set which belongs to its obligations and authorities (can be empty set);

Self-examining: Vm-set×Contract-set□Ppn-set. Based on the signed sc (∈ Contract-set), each vm (∈ VM -set) examines the performance status of its performing norm, and self-examining (vm, sc) = self-executing (vm, sc);

Inter-reporting: Vm-set × Contract-set□Ppn-set. Based on the signed sc (∈ Contract-set), each vm (∈ VM -set) reports the execution status of performing norm to collaboration partners, and Inter-reporting (vm, sc) = Self-executing (vm, sc);

Inter-examining:Vm-set×Contract-set□Ppn-set. Based on the signed sc (∈ Contract-set), each vm (∈ VM-set) examines the performing norm of service provider/demander, and Inter-examining (vm, sc) ∪ Self-examining (vm, sc) = pns(c)，Inter-examining (vm, sc) ∩ Self-examining (vm, sc)=□.

3.5.VO adaptive and self-evolutionary mechanism (VAEM)

Although the service contract has set up remedial performing norm to deal with foreseeable exceptions of contract performance, it will still encounter some unforeseen exceptions which lead to abnormal termination of the contract. The maintenance activities for these exceptions will be undertaken by the self-evolutionary mechanism.

VO Adaptive and self-Evolutionary Mechanism (VAEM) is defined as the following 10-tuple:

VAEM=(CPC,AEP,CV-events,CVT-principles,CE-plans,CE-actions,Monitoring,Analyzing, Planning, Executing).

CPC: Service contract performing circumstance set {CPCsc1, CPCsc2, …, CPCscn}, CPCsci denotes performing circumstance sci of service contract i.

AEP: Adaptive and evolutionary policy set, AEP = mo-policies ∪ an-policies ∪ pl-policies ∪ ex-policies, where mo-policies, an-policies, pl-policies, ex-policies respectively denote monitoring, analysis, planning, implementation policy subset.

4.Adaptive and flexible evolvement process

The flexible evolvement process is divided into four phrases: collaborative scenario monitoring, scenario variation analysis, collaborative evolution planning and evolution plan executing.

4.1.Collaborative scenario monitoring

Monitoring(x, CPCSC) => ¬∃ mop(∈ mo-policies) ∧ Exception(x,mop,CPCSC) ∧ Create(x,CVE). Monitoring activities are composed by self-examination and mutual examination of executing status of service contract performing norm. Common or specific scenario monitoring policies mop (∈mo-policies) are used to discovery exceptions of CPCsc(∈CPC), and then establish corresponding event of default CVE(∈CV-events).

4.2.Scenario variation analysis

Analyzing(x,CPCSC,CVE)=> ∃ anp(∈ an-policies) ∧ Compl(x,anp,CPCSC,CVE) ∧ Create(x,cvtp). Specific scenario variation analysis policy anp(∈an-policies), which is activated by CVE(∈CV-events), is used to analyze CPCsc(∈CPC), and provide analysis results cvtp(∈cvt-principles) (disposal principles of event of default).

4.3.Collaborative evolution planning

Planning(x,cvtp)=>∃plp (∈ pl-policies) ∧ enabled(cvtp,plp) ∧ Create(x,cep). Specific collaborative evolution planning policy plp (∈pl-policies), which is activated by disposal principles of event of default cvtp (∈cvt-principles), is used to program and generate evolution plan cep (∈ce-plannes).

4.4.Evolution plan executing

Executing(x,cep)=> ∃exp (∈ ex-policies) ∧ enabled(cep,exp) ∧ starting(x,ceas). Specific evolution plan executing policy exp (∈ex-policies), which is activated by evolution plan cep (∈ce-plannes), is used to start evolution action ceas (∈ce-actions) specified by exp.

5.Case study

This section applies a case of geospatial information services flow to verify effectivity of the proposed VO self-evolution model. The case is used to aid decision-making for flood emergency decision, in which, the VO established by flood emergency decision service provider FDS needs to integrate multiple geospatial information services (including WCS, WFS, WPS). The collaborative process of VO is shown in Figure 2.

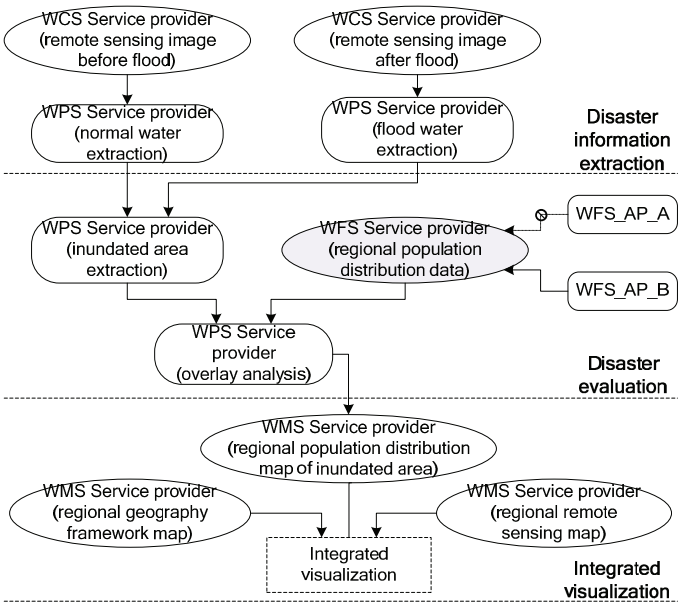

Fig. 2 Collaborative process of VO for flood emergency decision

Suppose APC:（Contract_info, Data_items, pn）has been established through consultation between FDS and regional population distribution data service provider WFS_AP. It means that data service provider signs commitment to provide datum service specified by Data_items to FDS in accordance with pn (contract performing norm set).

pn={(Norm001, contract inuring, FDS, pay 50$, in 3 days),

(Norm002, contract Performance, Norm001, Success,WFS_AP, open download Service, in 8 hours),

…

(Norm070, contract Performance, Norm002, Fail or Abnormity, FDS, CancelContract, Now),

(Norm071, contract Performance, Norm002, Fail or Abnormity, FDS, PostponDepositProvide, Now)}

These norms represent the following meanings:

Norm001, FDS is duty-bound to pay $50 deposit to WFS_AP within 3 days after execution of APC contract.

Norm002, FDS is duty-bound to open the network service within 8 hours after successful execution of Norm001.

Norm070, FDS has the right to immediately cancel APC contract when an execution exception is thrown by Norm002.

Norm071, FDS has the right to delay margin payment when an execution exception is thrown by Norm002.

Self-evolution reasoning polices for FDS are as follows:

- Scene Monitor Policy (SMP): {(first, time), (second, reputation)}.
- Scene Analysis Policy (SAP): {(time limited, penalty), (time limited and reputation of WFS_AP <7, cancellation), (time unlimited and reputation of WFS_AP<6, contract cancellation), (time unlimited and reputation of WFS_AP>6, renegotiation)}.
- Cooperation Evolvement Policy (CEP): {(penalty, penalty starting), (renegotiation, renegotiation starting), (contract cancellation and having person, changing person), (contract cancellation and having path, changing path), (contract cancellation and having no path, process thing left)}.
- Evolvement Plan Deployment Policy (EPDP): {(penalty starting, notify counter partner to pay penalty), (renegotiation starting, pause every related sub operation and start negotiation module), (changing person, pause every related operation and start changing person operation), (changing path, pause every related operation and start changing path operation), (process things left, pause every related operation and notify up operation)}.

We assume that FDS pays 50$ in 3 days after the execution of contract, but WFS_AP does not provide data service in 8 hours, thus a WFS_AP happens, which FDS must handle.

Firstly, FDS captures an exception in performing Norm002 with SMP (first, time), and finds that WFS_AP does not provide data service in 8 hours, so FDS concludes that it must cancel the APC by SAP(time limited and reputation of WFS_AP <7, cancellation). According to path generation policy, FDS find that there exists another WFS_AP provider, thus produces the evolvement plan: changing provider. Finally, according to plan deployment policy, FDS replaces the WFS_AP_A with WFS_AP_B and continues the cooperation. Thus the whole process of VO evolvement is over.

6.Conclusions

This paper presents a model of flexible virtual organization evolving facilitated by contract performing scene facilitated by policies and contracts, to handle exception event happened in VO. The model introduces the concept of policy, contract, and norm. Policy is the dynamic constraint to agent behavior, which can regulate the deducing behavior of autonomous agent. Contract is the constraint to agents involve in VO. The introduction of policy separates the management logic from application logic, which advances the applicability of the system. This paper defines policy attempt operator as an extension of attempt operator, and describes the flexible evolvement process after the happening of exception.

However, there is a great deal of further work required to make the policies and contract facilitated exception handling model more comprehensive, including the autonomic configure of policy, the importing of norm to regulate the behavior of individual agent, the improvement of contract description, etc.

References

[1]M. Stal, "Using architectural patterns and blueprints for service-oriented architecture," IEEE Software, vol. 23, no. 2, pp. 54-61, 2006.

[2]T. Erl, Service-oriented architecture: Concept, technology, and design, Professional Technical Reference, Upper Saddle River: Prentice Hall, 2005.

[3]T. Erl, Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services, London: Pearson Education, 2004.

[4]H. Afsarmanesh, M. Camarinha, "A Framework for Management of Virtual Organization Breeding Environments," In Proceedings of PROVE'2005, pp. 35-48, September 2005.

[5]L. Yu, L. Herman, Y. Stanley, "Adaptive Grid Service Flow Management: Framework and Model," In Proceedings of ICWS'2004, 2004.

[6]B. Yathiraj, P. Munindar, "Contract Enactment in Virtual Organizations: A Commitment-Based Approach," Proceedings of the 21st National Conference on Artificial Intelligence (AAAI), Boston: AAAI Press, pp. 722-727, July 2006.

[7]A. Mouaddib, L. Jeanpierre, "Dynamic Coalition of Resource-Bounded Autonomous Agents," In Proceedings of ICTAI'2007, pp. 117-123, 2007.

[8]M. Golani, A. Gal, "Optimizing Exception Handling in Workflows Using Process Restructuring," Business Process Management, pp. 407-413, 2006.

[9]B. Liao, J. Gao, J. Hu, J. Chen, "Ontology-Based Conceptual Modeling of Policy-Driven Control Framework: Oriented to Multi-agent System for Web Services Management," Lecture Notes in Computer Science, pp. 346-356, Springer-Verlag, 2004.

[10]Y. Zuo, B. Panda, "Information Trustworthiness Evaluation Based on Trust Combination," In Proceedings of the SAC'2006, pp. 1880-1885, 2006.

[11]S. Fatima, M. Wooldridge, N. Jennings, "Multi-issue negotiation with deadlines," Journal of Artificial Intelligence Research, vol. 27, pp.381-417, 2006.

[12]E. Tuci, "Cooperation Through Self-Assembly in Multi-Robot Systems," ACM Transactions on Autonomous and Adaptive Systems, vol. 1, no. 2, pp. 115-150, December 2006.

[13]R. Barrett, "People and Policies, Transforming the Human-Computer Partnership," In Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04), 2004.

[14]D. Grossi, H. Aldewereld, F. Dignum, Ubi Lex Ibi Poena, "Designing Institutions," In Proceedings of AAMAS'2006, Hakodate, Japan, May, 2006.